A sparse implementation of dynamic competition in continuous neural fields

Jean-Charles Quinton and Bernard Girau

Abstract This paper introduces a sparse implementation of the Continuum Neural Field Theory, promoting a trade-off in accuracy for higher computational efficiency and alleviated constraints on the underlying model. The sparse version reproduces the main properties of previous discrete 2D implementations, such as dynamic competition leading to localized focus activity or robustness to noise and distracters, with a much higher computational speed on standard computer architectures.

1 Introduction

Under adequate conditions, neural networks with lateral inhibition are able to maintain bubbles of activity in response to input excitation [12, 1, 10]. The lateral connectivity pattern classically takes the form of a difference of Gaussians function (DoG) with local excitation and large-scale inhibition. These developments fall under the Continuum Neural Field Theory (CNFT) and general convergence results have been reproduced extensively in experimental studies. Biologically inspired by the two dimensional topology of the cortical sheet, analyzed in terms of cortical maps and further decomposed in cortical columns [2], simulations were limited to two dimensions and most of them use a discretized version of the continuous equations. Anyway, from a computational point of view, updating the activity of a regular mesh of units with massive lateral connectivity has a polynomial algorithmic complexity, making any attempt to simulate a high dimensional CNFT difficult.

The bubbles emerging from such networks, i.e. localized Gaussian-like patches of activity, are able to focus on spatiotemporally coherent stimuli, despite the presence of noise or distracters [9]. Dynamic competition between units and their ex-

Jean-Charles Quinton, e-mail: jeancharles.quinton@loria.fr,

Bernard Girau, e-mail: bernard.girau@loria.fr,
Loria Laboratory, Campus Scientifique, B.P. 239, 54506 Vandoeuvre-lès-Nancy Cedex, France

citation by topologically organized inputs thus lead to the emergence of a robust attentional property. This property is quite generic and should be beneficial to any distributed system where decision or action is required.

1.1 A sparse implementation of the CNFT

For most experimental conditions, the focus map either rapidly converges to a global close-to-zero activity or to a set of distant bubbles. These bubbles must indeed be separated enough for the combined effect of their self-maintenance and input excitation to counterbalance the large-scale inhibition. In light of these observations, its seems reasonable to approximate the focus map by a sum of Gaussian fields, at least after the few steps needed for convergence. The key point of the model presented in this paper is to approximate any activity bubble by a Gaussian field, whose profile is determined by the Mexican hat function and CNFT parameters. The mesh of units used for each map in the standard discrete implementation is thus replaced by a set of Gaussian parameters (see Fig. 1).

As units generally rely on synaptic connections to modulate their activity and take fixed positions in topological neural networks, whether artificial or biological, the location parameter of the Gaussian distributions may similarly be constrained to take a finite set of values. This constraint is however not at all required by the so-called *sparse implementation* presented in the paper, which is spatially continuous. It could be anyway introduced to simulate the CNFT dynamics on Gas-nets models where Gaussian like diffusion patterns propagate activity between distant units [5]. These remarks however suppose that the Gaussian fields actually map physical units, whereas they may simply model the activity over the field, independently of the substrate.

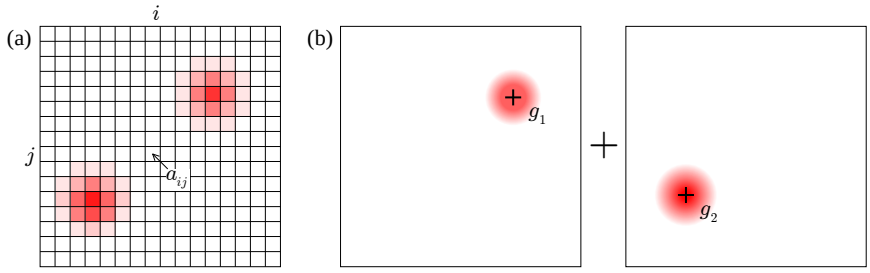

Fig. 1 Focus map representations when two bubbles have emerged. *(a)* In the discrete version, as a regular grid of units with activity $(a_{ij}) \in [0;1]^{n\times n}$ where n is the size of the grid. *(b)* In the sparse version, as the sum of Gaussian fields g_k defined by the parameters $(\mathbf{x}_k, I_k) \in ([-0.5;0.5]^2 \times [0;1])^k$ where k is the number of fields.

1.2 Expected benefits

The goal of a sparse implementation of the CNFT is twofold. First, alleviating the constraints on the underlying network makes the CNFT compatible with any system where topologically organized distributed processes compete for decision or action. At the same time, a sparse implementation allows a dramatic speed up in the computations. Fast simulation of the network dynamics may be useful to easily tune the parameters of the model. Moreover, real-time processing of high dimensional networks is a requirement when manipulating sensorimotor systems. When controlling robots for instance, time constraints are inherent to the interaction loop and must be dealt with.

To further develop the potential benefits from manipulating compact representations, details of the perception-action loop must be stated. Both biological and artificial systems interact with their environment through a large number of sensory and motor dimensions. At least in artificial systems, many sensory inputs and motor commands take a unique and precise value (joint angles for example). When using cortical map representations, single values must be projected on discrete 2D maps, the activity at each unit of each map regularly updated, before synthesizing the unit activities on motor maps into a single command to be sent to effectors. Although this might be totally irrelevant for biological systems, using compact representations of the activity on the different maps not only avoids such conversions, but also allows the manipulation of high dimensional spaces, without using self-organizing maps to reduce the dimensionality [7] and by limiting the combinatorial explosion of nD maps (with $n > 2$).

At a theoretical level, turning to an implementation with continuous inputs and bubbles makes several aspects of the analysis closer to the case described in the original CNFT equations [1]. A fixed spatial discretization is no longer required and the processing resolution can freely adapt to the task performed by the simulated system. Nevertheless, a fully continuous model apparently lacks the biological plausibility of discrete cortical maps with interconnected localized units. However, this might simply reflect a shift in focus from the underlying substrate to the map activity dynamics. In this perspective, it would remain useful as a practical tool, whatever its flaws or lack of accuracy for fine-grained models of cortical networks.

2 From discrete to sparse CNFT

In the following sections, we will adopt the notations introduced by Amari [1] and Rougier et al. [9], whose discrete implementation will serve as a reference.

2.1 Standard equations and discrete implementation

The focus neural field is represented by a manifold M in bijection with $[-0.5,0.5]^2$ and the membrane potential at the position vector $\mathbf{x}$ and time t on this field by $u(\mathbf{x},t)$. Similar notations are used for the input stimulation $s(\mathbf{x},t)$. The dynamics of the membrane potential is described by the following single-layer field equation of lateral inhibition type:

$$\begin{array}{l}\tau\frac{\partial u(\mathbf{x},t)}{\partial t} = -u(\mathbf{x},t) + \int_{x'\in M} w(\mathbf{x},\mathbf{x}')u(\mathbf{x}',t)d\mathbf{x}' \\ \qquad\qquad +s(\mathbf{x},t)+h\end{array} \tag{1}$$

where h is the resting potential and $w(\mathbf{x},\mathbf{x}')$ the lateral connection weight function satisfying the following equation:

$$w(\mathbf{x},\mathbf{x}') = Ae^{\frac{|\mathbf{x}-\mathbf{x}'|^2}{a^2}} - Be^{\frac{|\mathbf{x}-\mathbf{x}'|^2}{b^2}} \tag{2}$$

In order to perform numerical simulations of the CNFT dynamics, the continuous equations have been discretized as follows:

$$\begin{array}{l}\tau\frac{\partial u(\mathbf{x}_{ij},t)}{\partial t} = -u(\mathbf{x}_{ij},t) + \frac{1}{n^2}\sum_{(k,l)\in[0;n[^2} w(\mathbf{x}_{ij},\mathbf{x}_{kl})u(\mathbf{x}_{kl},t) \\ \qquad\qquad +s(\mathbf{x}_{ij},t)+h\end{array} \tag{3}$$

where $\mathbf{x}_{ij}$ is the discrete position associated to the unit (i,j) and given by Eq. 4.

$$\mathbf{x}_{ij} = \left(\frac{i}{n}-0.5,\frac{j}{n}-0.5\right)\forall(i,j)\in[0;n[^2 \tag{4}$$

2.2 Sparse implementation

To exclusively manipulate Gaussians in a sparse implementation, all terms in Eq. 1 must be transformed accordingly. At this point, it may be argued that stimulations $s(\mathbf{x},t)$ will not necessarily take the form of normal distributions, especially when considering noisy inputs or overlapping stimuli. However, any discrete signal can be approximated by a sum of Gaussians, their number being positively correlated with the precision required [4]. Although efficient algorithms to perform such a decomposition are available [3], one goal of the current implementation is to make it compatible with sparse inputs.

In artificial systems, these inputs might be associated with bottom-up signals (coming from various measurement systems and modalities) as well as top-down signals competing for decision or action. In biological systems, sensory features are encoded by myriads of neurons, but population coding is often considered as it averages the variability of single neuron spike trains, even when each of them already takes inputs from large receptive fields [6]. Receptive fields have been found to be

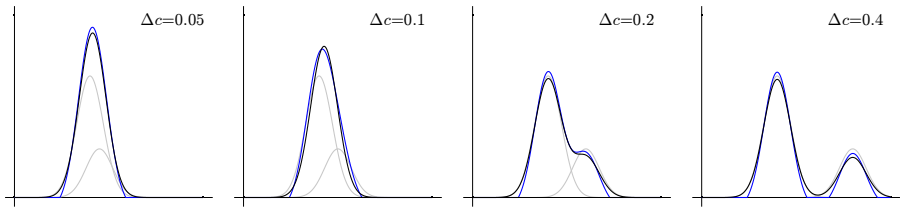

Fig. 2 Comparison of the discrete and sparse algorithms on a few iterations with increasingly more distant Gaussian components (*light gray*). The discrete algorithm (*dark gray*) computes an output activity for each unit (1000 units uniformly sampled in $[0;1]$). The sparse algorithm (*black*) computes the reciprocal inhibition between Gaussian distributions, merges them when close enough and outputs the sum of the resulting distributions. From left to right, the distance between the input centers Δc takes values in $\{0.05, 0.1, 0.2, 0.4\}$. The limit between excitation/merging and inhibition is 0.1 for both versions so that the maximal distortion appears for $\Delta c \simeq 0.1$.

approximately separable into a sum of amplitude modulated Gaussian components, for instance in the visual system [8, 11].

The effect of the CNFT computation on distributions whose centers are sufficiently separated relatively to the excitatory standard deviation a in Eq. 2 ($\Delta c >> a$), is only inhibitory. Their interactions can then be well approximated by a point to point interaction, only considering activity distribution centers (as centers of mass are used in point mechanics). This can be proved using Taylor developments of the competition function w between positions $\mathbf{x}$ and $\mathbf{x}' + \mathbf{dx}$ with $\mathbf{dx} \ll |\mathbf{x} - \mathbf{x}'|$. The same intuition is mathematically valid for $\Delta c = 0$, the sum of aligned normal distributions being a Gaussian distribution. Nevertheless, this approximation does not hold when Δc takes intermediate values. To evaluate the distortions resulting from such a rough approximation, numerical simulations have been performed. To get an idea of the potential quantitative differences between the discrete and sparse implementations for a range of Δc values, Fig. 2 compares the fields produced after a few iterations.

From now on, the following notations are adopted. A generic field G will be defined as the sum of k components g_k. Determined by the set of parameters $(\mathbf{x}_k, I_k)$, g_k denotes a Gaussian field of amplitude I_k centered on $\mathbf{x}_k$. Let $g_k(\mathbf{x})$ be the activity propagated by the Gaussian component g_k at the point $\mathbf{x}$ satisfying Eq. 5.

$$g_k(\mathbf{x}) = I_k * e^{\frac{|\mathbf{x}_k - \mathbf{x}|^2}{\sigma^2}} \tag{5}$$

with σ being equal to the excitatory standard deviation a of Eq. 2 for the focus field components. In most cases, this produces the stereotypical profile of the focus bubbles and is coherent with the choice of synthesizing each bubble by a single Gaussian field. Finally, the potential at any point of the field can then be computed as in Eq. 6. The choice of a field named G in this section is arbitrary and G could be

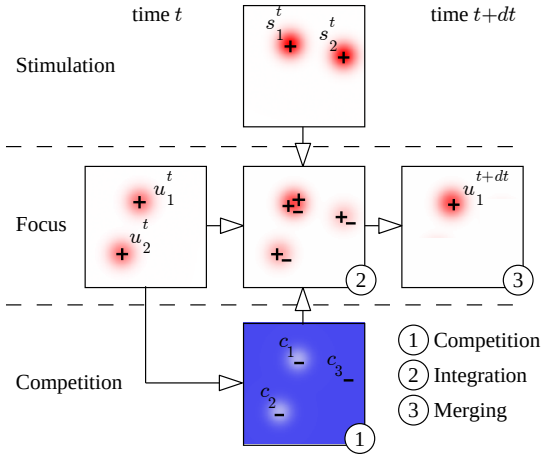

Fig. 3 Algorithmic decomposition of the sparse implementation. ① A competition field C is produced by propagating activity from $\{u_1^t, u_2^t\}$ to $\{u_1^t, u_2^t, s_1^t, s_2^t\}$. ② Components from the focus field $\{u_1^t, u_2^t\}$, input field $\{s_1^t, s_2^t\}$ and competition field $\{c_1, c_2, c_3\}$ are integrated. ③ Close components are merged, and resulting components with negative intensity removed ($s_2^t \cup c_3$ and $u_2^t \cup c_2$). Only one component remains ($u_1^{t+dt} = u_1^t \cup s_1^t \cup c_1$), reflecting the convergence towards a single bubble of activity.

substituted by any field introduced in the equations and paragraphs to come, such as the stimulation field S or focus field U. Let S^t, U^t and s_k^t, u_k^t be their respective state and components at time t.

$$g(\mathbf{x},t) = \sum_k g_k^t(\mathbf{x}) \tag{6}$$

To briefly introduce the algorithmic decomposition of Eq. 1 detailed in the following sections, we illustrate the expected dynamics of the model on Fig. 3. For each temporal step of the simulation, a competition field C is computed by propagating activity from the different components of the focus field U (see Sec. 2.3). The input, focus and competition fields are then integrated in a single field where spatiotemporally coherent stimuli form clusters of excitatory components (see Sec. 2.4). Activity resulting from the presence of noise or distracters (such as s_2^t on the figure) is not reinforced by focus components and counterbalanced by inhibitory competition components (c_3). Finally, close components are merged (u_1^{t+dt}) and Gaussian with negative activity eliminated (see Sec. 2.5). In standard conditions, this ensures that a reduced set of distant bubbles will emerge, each represented by a single component.

2.3 Competition

The first step consists in computing an equivalent to the dynamic competition term in Eq. 1. For each focus and stimulation component, an activity I_k^c is computed by applying Eq. 7. For each u_k^t and s_k^t considered, an inhibitory component c_k of parameters $(\mathbf{x}_k, I_k^c)$ is thus produced. All c_k are combined in global inhibitory field C (for competition).

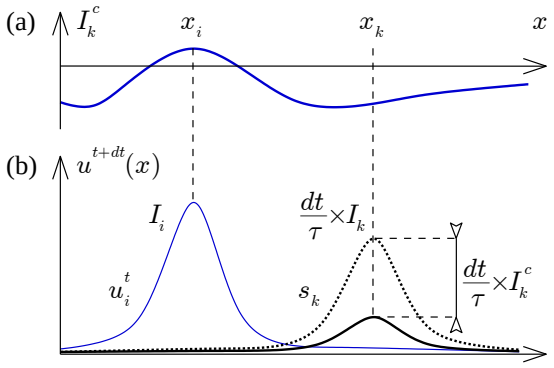

Fig. 4 Competition between components. *(a)* The inhibitory activity I_k^c is computed at $\mathbf{x}_k$ for the input field s_k (supposing there is only one Gaussian field component u_i on the focus). *(b)* The activity at $\mathbf{x}_k$ is reduced. If I_k^c had not been taken into account (dashed line), s_k would not have been negligible when computing u^{t+dt} although far away from the only active bubble on the focus u_i^t.

$$I_k^c = \frac{1}{n}\sum_{i=1}^{n} w(\mathbf{x}_k,\mathbf{x}_i)I_i \tag{7}$$

where n is the number of components on the focus field. The lateral competition weight function w has already been introduced in Eq. 2. Note that in the original equations, the dynamic competition was evaluated at each point of the field. In a sparse implementation, it is therefore required to account for the inhibition at any place where there is a Gaussian distribution. The stimulation S being independent of the internal computations performed, activity on the input field must also be inhibited (see Fig. 4). For example on Fig. 3, s_2^t would not be inhibited if c_3 was not introduced and only the interactions between u_1^t and u_2^t were taken into account.

2.4 Integration

The focus field U^t, inhibitory field C resulting from the lateral competition and input field S must then be combined and integrated over time to reproduce the dynamics of Eq. 1. This integrative step is described by the following equation:

$$U^{t+dt} = U^t \cup \frac{dt}{\tau}\left[\left(-U^t \cup C \cup S^t\right) + h\right] \tag{8}$$

where the $\cup$ operator applied to fields actually corresponds to the union of the component sets. When computing the potential at a given position by applying Eq. 6, this is actually equivalent to adding the contributions of the different fields. The scalar multiplication (by $\frac{dt}{\tau}$) and addition (of h) are directly applied to the intensity of the Gaussian components. If we respectively denote by U' and U'' the fields equivalent to $-U^t \cup C \cup S^t$ and $\frac{dt}{\tau}(U'+h)$, their component parameters $(\mathbf{x}_k, I'_k)$ and $(\mathbf{x}_k, I''_k)$ satisfy:

$$I''_k = \frac{dt}{\tau}(I'_k + h) \tag{9}$$

2.5 Merging

Although the potential at any point of the field can be updated and computed at the end of the previous section, yet another step is required to account for the bubbles reinforcement and to avoid a combinatorial explosion. The unions in Eq. 8 indeed lead to an ever increasing number of components on the focus field. Even though in practice and for efficiency purpose, computations are factored so that the number of Gaussian fields centered on the same position remains minimal, stimulations are not constrained and constantly inject new components into the focus field. If the number of input Gaussians at t is n_t, an underestimation of the number of components on the focus field at time T is $\sum_{t=0}^{T} n_t$, which is quite problematic when considering the polynomial cost of computing the lateral competition. A merging algorithm is therefore introduced.

Similar constraints appear in different research fields when manipulating large Gaussian mixture models or HMM-based models with continuous densities [13]. The main difference being that the actual distribution of local stimulations is not here accessible to the model. Due to the robustness of the CNFT dynamics and efficiency constraint of the implementation, a simple Euclidian distance between the distribution locations is used as the merging criterion. This distance threshold is chosen to match the excitatory standard deviation a as to facilitate and reinforce the emergence of stereotyped bubbles of activity.

Algorithm 1 Merging algorithm for the Gaussian components on the focus field

```
1.  %Find close Gaussian pairs
2.  P ← ∅
3.  for all (u_i, u_j) ∈ U^2
4.      if |x_i − x_j| < a
5.          P ← P ∪ (u_i, u_j)
6.      end
7.  end

In practice P is kept sorted as to easily select
and always merge the closest components in
the algorithm when choosing (u_i, u_j)

8.  %Iterate the merging on pairs
9.  while P ≠ ∅
10.     u_new ← merge(u_i, u_j)
11.     P ← P \ { pairs with u_i or u_j}
12.     U ← U ∪ u_new \ {u_i, u_j}
13.     for all u_i ∈ U
14.         if |x_i − x_new| < a
15.             P ← P ∪ (u_i, u_new)
16.         end
17.     end
18. end
```

With the standard equations 1 and 3, bubbles track the moving stimuli on which they are focused because the renewed inputs introduce an asymmetry in the dynamic competition computations. The shifted bubble is in turn reinforced by the input, although always lagging behind the stimulation. The same tracking characteristic is here achieved by the *merge* function introduced in Algorithm 1 and defined by Eq. 10.

$$\mathbf{x}_{new} = \frac{I_i}{I_i + I_j}\mathbf{x}_i + \frac{I_j}{I_i + I_j}\mathbf{x}_j$$

$$I_{new} = I_i + I_j - I_i \times I_j \times \frac{|\mathbf{x}_i - \mathbf{x}_j|^2}{\alpha^2} \tag{10}$$

where α is a constant higher than a. When components are perfectly aligned, as it happens when computing the inhibitory field I^- from the focus field U, the Gaussian amplitudes are simply added during the merging operation, which is coherent with the discrete unit based computations of Eq. 3. When a focus component and a stimulation are close enough for the merging to occur, their intensities have respectively been scaled by $1 - \frac{dt}{\tau}$ and $\frac{dt}{\tau}$. This weighting allows the bubble to slightly shift towards the new stimulus while reinforcing or at least maintaining its activity. Taking $\alpha = +\infty$ cancels the last term and introduces additional discontinuities but still allows the algorithm to perform correctly as long as a is chosen carefully. Components of negative amplitude at the end of the merging phase are simply removed from the field, thus limiting the focus potential to positive values.

Finally, when a single locus representing the entire focus field is needed, whether for undertaking actions or statistical analysis purpose, the barycenter $\mathbf{c}$ of the remaining components (i.e. those which have not been merged and represent distant bubbles of activity) can be rapidly computed by applying Eq. 11.

$$\mathbf{c} = \frac{\sum_k I_k \times \mathbf{x}_k}{\sum_k I_k} \tag{11}$$

3 Results and performance

The analysis of the model dynamics and its performance are relative to the competition and attentional properties of the CNFT. The discrete implementation described in [9] is used as a reference, and the same evaluation process is adopted. The authors commit to the assumption that a single bubble should emerge on the focus field and that it should robustly track the associated input stimulus despite noise and distracters. The scenarios posed by Rougier and Vitay guarantee that for adequate sets of parameters, there will be enough iterations for a single bubble to appear before noise and distracters are added. As a consequence, the performance is easily ascertained by measuring the distance between the input stimuli and focus bubble centers at any time.

3.1 Functional results

Although the discrete and sparse implementations share general principles and parameters, there are many qualitative differences in the algorithms, not to state the most obvious difference in the type of stimulation provided to the system. Since the two versions might not stand on equal footing, the goal here is mainly to illustrate

how a Gaussian based implementation may reproduce the classical CNFT characteristics.

A fair comparison at least requires using optimal sets of parameters, which might greatly differ between implementations. To avoid complex hand-tuning, genetic algorithms are applied to evolve a population of parameter vectors independently for each version. The fitness function to minimize is based on the cumulative error on a set of well chosen scenarios simulated in sequence. Indeed, depending on the task on which the optimization process is applied, optimal parameters may differ. For instance, the a value is highly correlated to the distance between the stimuli to focus on: a large value may improve the results only when unambiguous distant stimuli are presented. The parameters are thus optimized as to produce the best results for all three following scenarios:

A) 2 bell-shaped distant stimuli s_1 and s_2 are introduced at time $t = 0$. Their intensity are governed by $I_1 = 0.4$ and $I_2(t) = 0.5 + 0.5\cos(\pi \times (t/5))$.
B) 1 bell-shaped stimulus of standard deviation 0.1 and intensity 1.0 follow a circular trajectory of radius 0.2 around the point $(0,0)$ at 10 deg/s from $t = 0$. From $t = 1$, 5 distracters are added and take new random positions on the field every 1 s.
C) 1 bell-shaped stimulus (same as above). At $t = 1$, Gaussian noise of amplitude 0.5 is added.

In the discrete version, noise is easily added by changing the stimulation intensity by a random amount for each $\mathbf{x}_{ij}$. This is not possible in the sparse version as each Gaussian component already corresponds to a correlated activity over the entire field. Adding a large number of random components would not do the trick as it would be quite equivalent to adding distracters. However, once a bubble is stabilized as it is guaranteed by the experiments reproduced, units outside the bubble are largely inhibited and the associated inputs have no effect on further computations. Noise introduces asymmetries under the bubble and destabilize it, which is roughly

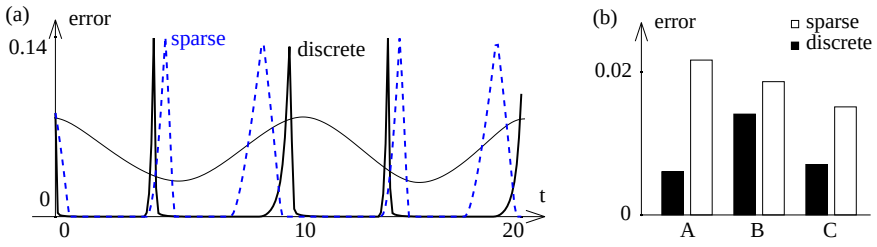

Fig. 5 Functional results. (*a*) Normalized error dynamics for scenario *A*. Although the shifts in attention associated with a high error occur at slightly different times, the same kind of hysteresis appears for the sparse (*dashed*) and discrete (*plain*) versions relatively to s_2 intensity (*cosine function*). (*b*) Although the error is lower for the discrete version, both implementations remain below 2% for the scenarios *A* to *C*.

reproduced here by randomly shifting the intensity of stimulations and translating their center. Results for scenarios A to C are shown and commented on Fig. 5.

3.2 Computational speed up

The following paragraphs do not take into account the cost of decomposing a discrete input into a sum of Gaussian distributions and are thus mainly relevant for artificial systems where inputs are compatible, as argued in Sec. 1.2 and 2.2. A simple algorithmic complexity analysis helps understanding the computational benefits from the sparse implementation. Let n^2 be the size of the maps. Using infinite asymptotics, all terms are dominated by the convolution computed for the lateral competition, so that $T(n) = O(n^4)$ using Landau notation[1]. When using a singular value decomposition (SVD) on the lateral connection kernel, the 2D convolution can be decomposed in two orthogonal 1D convolution and the complexity drops to $T(n) = O(n^3)$. The initial cost of computing the SVD can be neglected as long as the kernel remains constant throughout the entire simulation. When modeling higher dimensional maps of size n^m with $m > 2$ and considering the non optimized version, the complexity becomes $T(n) = O((n^m)^2)$ and projecting the data on the low dimensional space is required.

For the sparse implementation again, most computations can be neglected relatively to the inhibition and merging operations. The complexity of the lateral inhibition is in the worst case $T_c(n_u) = O({n_u}^2)$ if we let n_u be the number of components on the focus field, supposed far greater than the number of input components. The complexity of the merging operation depends on the actual distribution of the components, and can therefore hardly be mathematically approximated. In practice, and as soon as a group of components get reinforced by the inputs, the lateral competition rapidly inhibits a large number of components and their number drastically decreases. In such a case, infinite asymptotics is no more valid but the computational cost of updating the fields anyhow drops far below the cost of the discrete version, that only depends on the size of the map. This is not surprising as making localized bubbles of activity emerge is the sole goal of the lateral competition, and was one reason for developing a sparse implementation. Experimental performance results are reproduced on Table 1 for different implementations and scenarios.

4 Conclusion

In this paper, we introduced a sparse version of the CNFT able to reproduce its main properties and speed up the computations with a relative independence to the number of dimensions manipulated. The strong assumptions and rough approximations introduced in the process must however be taken into account when applying it to in-

[1] Although n does not usually take very high values, a map of size 50×50 is common and already justifies the effect of n on performance.

Table 1 Comparison of mean computation time (in μs) to update the model for one timestep, exclusive of the input generation. Although there is a bigger variance with the sparse implementation since the computations depend on the number and distribution of components, the time needed remains over 50 times lower than for the SVD optimization of the discrete version of the model.

Scenario	Discrete version	Discrete (SVD)	Sparse version
A (alternation)	321131	37537	596
B (distracters)	321084	37724	953
C (noise)	321003	37491	632

puts not easily reduced to a sum of Gaussian components. Part of the problem might then be deviated to the input filtering and decomposition algorithms, instead of being treated directly by the CNFT. Perspectives include a fully distributed implementation for real-time high dimensional robotics applications, since the reduced number of components, their limited exchanges and low memory requirements make it compatible with parallel hardware that provides fast paced computations but limited resources.

References

1. S.-I. Amari. Dynamics of pattern formation in lateral-inhibition type neural fields. *Biological Cybernetics*, 27(2):77–87, 1977.
2. Y. Burnod. *An adaptive neural network: The cerebral cortex*. Masson, 1989.
3. J. Childs, C.-C. Lu, and J. Potter. A fast, space-efficient algorithm for the approximation of images by an optimal sum of gaussians. In *Graphics Interface*, pages 153–162, 2000.
4. A. Goshtasby and W. D. O'Neill. Curve fitting by a sum of gaussians. *CVGIP. Graphical models and image processing*, 56(4):281–288, 1994.
5. P. Husbands, T. Smith, N. Jakobi, and M. O'Shea. Better living through chemistry : Evolving gasnets for robot control. *Connection Science*, 10(3-4):185–210, 1998.
6. E. Kandel, J. Schwartz, and T. Jessell. *Principles of Neural Science*. McGraw-Hill, New York, 2000.
7. O. Ménard and H. Frezza-Buet. Model of multi-modal cortical processing: coherent learning in self-organizing modules. *Neural Networks*, 18(5-6):646–55, 2005.
8. R. Rodieck. Quantitative analysis of cat retinal ganglion cell response to visual stimuli. *Vision Research*, 5(11):583601, 1965.
9. N. P. Rougier and J. Vitay. Emergence of attention within a neural population. *Neural Netw.*, 19(5):573–581, 2006.
10. J. Taylor. Neural bubble dynamics in two dimensions: Foundations. *Biological Cybernetics*, 80:5167–5174, 1999.
11. T. Wennekers. Separation of spatio-temporal receptive fields into sums of gaussian components. *Journal of Computational Neuroscience*, 16(1):27–38, 2004.
12. H. R. Wilson and J. D. Cowan. A mathematical theory of the functional dynamics of cortical and thalamic nervous tissue. *Kybernetic*, 13:55–80, 1973.
13. W. Xu, J. Duchateau, K. Demuynck, and I. Dologlou. A new approach to merging gaussian densities in large vocabulary continuous speech recognition. In *IEEE Benelux Signal Processing Symposium*, pages 231–234, 1998.